Recap

1. Introduction

L'objectif de cet exposé est de présenter les travaux pratiques réalisés dans le cadre du cours de Programmation Impérative. Plusieurs exercices pratiques ont été complétés, couvrant diverses séries de problèmes qui illustrent des concepts clés en programmation C, tels que la gestion de la mémoire, la manipulation des listes chaînées, et la création de bibliothèques dynamiques.

2. Séries de Problèmes

Série cx18

1. **cx18.1 : Affichage d'une liste en ordre inverse**
   - Fonction `putlist()` réécrite pour afficher les éléments d'une liste chaînée en ordre inverse en utilisant un algorithme de pile.
2. **cx18.6 : Création d'une bibliothèque dynamique**
   - Les fonctions de gestion des listes ont été extraites et placées dans une bibliothèque dynamique. Cela inclut des opérations telles que la construction et l'inversion de listes.

Série cx15

1. **cx15.3 : Lecture d'un fichier texte**
   - Création d'un fichier texte contenant des chaînes de caractères. Un programme lit ces chaînes et les stocke dans une table pour les afficher ensuite.

Série cx17

1. **cx17.2 : Exploitation d'une STOPLIST**
   - En utilisant la solution précédente de cx15.3, une liste de mots "interdits" (STOPLIST) a été ajoutée au programme, permettant de filtrer les mots lors de la lecture des données.
2. **cx17.6 : STOPLIST avec une liste élastique**
   - Adaptation du programme pour utiliser une STOPLIST représentée sous forme de liste élastique (liste chaînée), permettant une flexibilité dans la gestion dynamique des mots à exclure.
3. **cx17.7 et cx17.8 : Utilisation de bibliothèques dynamiques**

- Les solutions précédentes ont été adaptées pour utiliser les bibliothèques dynamiques créées dans cx18.6 et cx18.7, facilitant la modularité et la réutilisation du code.

Série cx25

1. **cx25.0 : Émulateur de lecture numérique**
   - Création d'un petit programme qui simule un processeur avec une mémoire de taille fixe. Ce programme lit un fichier contenant des instructions et exécute ces instructions, affichant les états de l'accumulateur (A) et du compteur de programme (PC).
2. **cx25.1 : Boucle d'évaluation en mode STEPPER**
   - Amélioration de l'émulateur avec une boucle d'évaluation interactive, où chaque instruction nécessite une confirmation de l'utilisateur avant de passer à l'instruction suivante. Ce mode permet de déboguer et de suivre l'exécution du programme étape par étape, similaire à un mode "stepper" dans les débogueurs.

3. Conclusion

Ce parcours à travers divers exercices de programmation impérative a permis de renforcer la compréhension des concepts de base en C, notamment la manipulation des structures de données dynamiques, la gestion de la mémoire et l'organisation modulaire du code.

Résumé des Concepts Clés pour les Exercices de Programmation Impérative

1. cx18.1 : Affichage d'une Liste en Ordre Inverse

- **Concept Principal :** Manipulation des listes chaînées.
- **Compétence Clé :** Parcourir une liste chaînée et inverser son ordre. Cela nécessite de comprendre comment chaque élément de la liste pointe vers le suivant.
- **Points à retenir :**
  - Utilisation de la récursivité pour parcourir la liste.
  - Empiler les éléments dans l'ordre de leur apparition et les afficher dans l'ordre inverse.
  - Allocation dynamique de mémoire pour les nœuds avec `malloc`.

2. cx18.6 : Bibliothèque Dynamique pour la Gestion des Listes

- **Concept Principal :** Création et utilisation de bibliothèques dynamiques.

- **Compétence Clé :** Modularisation du code en déplaçant les fonctions dans une bibliothèque partagée pour les réutiliser dans différents programmes.
- **Points à retenir :**
  - Créer une bibliothèque dynamique (`.so` file) à partir de code C.
  - Compiler un programme en liant la bibliothèque dynamique.
  - Utilisation de la commande `gcc` avec l'option `-shared` pour créer des objets partagés.
  - L’utilisation de la variable d'environnement `LD_LIBRARY_PATH` pour spécifier le chemin de la bibliothèque lors de l'exécution.

3. cx15.3 : Lecture de Fichier Texte dans une Table de Chaînes

- **Concept Principal :** Manipulation des fichiers en C.
- **Compétence Clé :** Lire des données à partir d'un fichier texte et les stocker dans un tableau de chaînes.
- **Points à retenir :**
  - Utilisation des fonctions `fopen`, `fscanf`, et `fclose` pour lire un fichier texte.
  - Stocker les chaînes de caractères dans un tableau en utilisant un buffer pour chaque chaîne.
  - Gérer les erreurs lors de l’ouverture de fichiers et de la lecture de données.

4. cx17.2 : STOPLIST

- **Concept Principal :** Filtrage de données à l'aide d'une liste de mots interdits.
- **Compétence Clé :** Comparer les mots d'un fichier de données avec une liste de mots interdits (STOPLIST) et filtrer ceux qui correspondent.
- **Points à retenir :**
  - Rechercher une chaîne dans un tableau de chaînes en utilisant `strcmp`.
  - Combiner les résultats de la lecture d'un fichier avec un processus de filtrage.

5. cx17.6 : STOPLIST avec une Liste Élastique

- **Concept Principal :** Utilisation de structures de données dynamiques.
- **Compétence Clé :** Représenter la STOPLIST comme une liste chaînée au lieu d'un tableau statique pour permettre une gestion plus flexible des données.
- **Points à retenir :**
  - Implémenter une liste chaînée où chaque nœud contient un mot de la STOPLIST.
  - Ajouter des mots à la liste de manière dynamique en allouant de la mémoire au fur et à mesure.

6. cx17.7 et cx17.8 : Utilisation de Bibliothèques Dynamiques pour STOPLIST

- **Concept Principal :** Réutilisation de code à travers des bibliothèques dynamiques.
- **Compétence Clé :** Intégrer des fonctions dynamiques dans des programmes qui utilisent des structures de données avancées.
- **Points à retenir :**
  - Les fonctions de gestion de la liste STOPLIST sont déplacées vers une bibliothèque dynamique, facilitant leur utilisation dans différents projets.
  - Le processus de compilation implique le lien avec la bibliothèque créée.

7. cx25.0 : Émulateur pour la Lecture de Code Numérique

- **Concept Principal :** Simulation d'un processeur simple.
- **Compétence Clé :** Lire et exécuter des instructions numériques stockées dans un fichier, avec un ensemble de commandes simples (addition, soustraction, etc.).
- **Points à retenir :**
  - Charger un programme en mémoire à partir d’un fichier texte.
  - Simuler un accumulateur (registre) et un compteur de programme (PC) pour suivre l’exécution des instructions.
  - Implémenter des instructions en utilisant un switch-case pour différentes opérations.

8. cx25.1 : Boucle d'Évaluation Interactive (STEPPER)

- **Concept Principal :** Exécution interactive des instructions avec une boucle de contrôle.
- **Compétence Clé :** Créer un débogueur interactif qui exécute les instructions une par une, demandant une validation utilisateur avant de passer à l'instruction suivante.
- **Points à retenir :**
  - Attendre l'entrée utilisateur avec `getchar` pour créer un mode stepper.
  - Simuler les étapes d'exécution avec affichage des états intermédiaires du compteur de programme et de l'accumulateur.

Récapitulatif des Points Importants

- **Manipulation de Listes Chaînées :** Compréhension des structures de données dynamiques et de leur gestion en mémoire.

- **Bibliothèques Dynamiques :** Modularisation du code et création de bibliothèques partagées pour réutiliser les fonctions dans différents programmes.
- **Lecture et Écriture de Fichiers :** Manipulation de fichiers en C pour lire des données structurées et les traiter.
- **Simulation de Processeur :** Utilisation de la mémoire, des registres et des instructions pour simuler un processeur simple.
- **Débogage Interactif :** Implémentation d'un mode pas-à-pas pour suivre l'exécution des instructions.